

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Edward Seeman
Appl. No.: 10/828,517
Filed: April 19, 2004
Docket No.: 2057
Conf. No.: 4284
Title: **METHOD, SYSTEM, AND COMPUTER-READABLE MEDIUM
FOR CREATING ELECTRONIC LITERARY WORKS,
INCLUDING WORKS PRODUCED THEREFROM**

Art Unit: 2176
Examiner: James J. Debrow

Action: **DECLARATION UNDER 37 C.F.R. §1.131**
Date: September 24, 2008

TO: Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Ed Seeman, a named inventor in the above-identified application, declares as follows:

1. My name is Ed Seeman and I reside in Arvada, Colorado. I am making this declaration in support of the above-identified application, for which I am one of the inventors.

2. I understand that the Examiner is rejecting the pending claims of this application, at least in part, based on the teachings of publication number US 2007/0011607 A1 to Lazareck et al. ("Lazareck"). It is also my understanding that the Lazareck reference has an effective filing date of February 7, 2003.

3. The present application has been assigned by myself and the other inventors to Real-E Studios, LLC ("Real-E. Studios"). I have been a member of Real-E Studios since its inception.

4. Attached as Exhibit A to this declaration is a "General Contract for Software Development", which was entered into on November 18, 2002 between Real-E Studios and an independent contractor whose identity has been redacted for confidentiality reasons. Other areas of the software development agreement which are not relevant to this declaration have also been redacted for similar reasons. The software development contract was accompanied by a project description which is attached as Exhibit B to this declaration. This project description describes in some detail the invention that was applied for in the above-identified application. The various features described in the project description were developed in the United States.

Exhibit B particularly describes at least the following features relating to a computerized method for creating an electronic literary work on a computer system that includes a display:

- a. providing an archive image set which includes a plurality of digital images;
- b. populating a first working area of an application program with selected digital images from the archive image set, thereby to generate a contact sheet view comprising an imported image set;
- c. populating a second working area of the application program with selected digital images from the imported image set, thereby establishing a working image set;

d. arranging the digital images which comprise the working image set into a selected storyboard sequence;

e. associating respective image data with at least some of the digital images in the working image set, thereby to define pairs of companion image items; and

f. displaying the pairs of companion image items on the display according to the storyboard sequence.

5. In my opinion, Exhibit B discloses the subject matter claimed in the above-identified application. Furthermore, Exhibit B was prepared prior to February 7, 2003 and accompanied the software development agreement that was entered into prior to February 7, 2003.

6. The declarant further states that the above statements were made with the knowledge that willful false statements and the like are punishable by fine and/or imprisonment, or both, under section 1001 of title 18 of the United States code, and that any such willful false statement may jeopardize the validity of this application or any patent resulting therefrom.

Dated this 24TH day of September, 2008

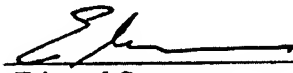

Edward Seeman

Exhibit A

GENERAL CONTRACT FOR SOFTWARE DEVELOPMENT

This Contract for Software Development (this "Contract") is made effective as of November 18, 2002, by and between Reale Studios, LLC of 9911 W 86th Pl, Arvada, Colorado 80005, and [REDACTED]. In this Contract, the party who is contracting to receive the services will be referred to as Reale, and the party who will be providing the services will be referred to as [REDACTED].

1. DESCRIPTION OF SOFTWARE DEVELOPMENT.

Beginning on November 18, 2002, [REDACTED] will provide the software development services described in the attached exhibits as they pertain to Version 1.0 of the RealeBook Reader/Writer. (Collectively, the "Project"):

2. PAYMENT FOR THE PROJECT.

Reale will pay compensation to [REDACTED] for the Project in the amount of \$60/hour.

The maximum amount paid for the Project will not exceed \$14,400. If the total amount owed to [REDACTED] equals \$14,400 prior to completing the Project described in 1) above, [REDACTED] will notify Reale that there will be a cost overrun. [REDACTED] will then provide Reale with an estimate of the cost required for completion of the Project. [REDACTED] will not resume working on the Project until Reale has agreed to the final cost to complete the Project and has authorized completion of same. Payment schedule will be as follows:

\$5000 on 15 December

\$5000 w/in seven days after delivery of initial complete build (approximately 10 January)
remainder on delivery of build w/ required fixes, installer, and source code

3. TERM/TERMINATION.

This Contract will terminate automatically upon completion by [REDACTED] of the Project required by this Contract.

4. WORK PRODUCT OWNERSHIP.

Any intellectual property, copyrightable works, ideas, discoveries, inventions, patents, products, or other information (collectively the "Work Product") developed in whole or in part by [REDACTED] in connection with the Project will be the exclusive property of Reale.

5. DOCUMENTATION.

Except as otherwise agreed to by the Parties, all documentation developed by [REDACTED] for the Project shall be and will remain the exclusive property of Reale. [REDACTED] hereby assigns to Reale all rights in and interests to the Documentation. As used herein, "Documentation" includes all materials, work papers, configurations, manuals and other work product prepared by or on behalf of [REDACTED] or otherwise used by [REDACTED] its subcontractors or agents in connection with the development of the Project. This shall not include products wholly owned by [REDACTED] that are commonly available for free or purchase, and are not necessary for the installation or function of the delivered product.

6. CONFIDENTIALITY.

[REDACTED] and its employees, agents, or representatives will not at any time or in any manner, either directly or indirectly, use for the personal benefit of [REDACTED] or divulge, disclose, or communicate in any manner any information that is proprietary to Reale. [REDACTED] and its

employees, agents, and representatives will protect such information and treat it as strictly confidential. This provision will continue to be effective after the termination of this Contract.

Upon termination of this Contract, [REDACTED] will return to Reale all records, notes, documentation and other items that were used, created, or controlled by [REDACTED] during the term of this Contract for the purpose of completing the contracted work.

7. INDEMNIFICATION.

[REDACTED] agrees to indemnify and hold harmless Reale from all claims, losses, expenses, fees including attorney fees, costs, and judgments that may be asserted against Reale that result from the acts or omissions of [REDACTED] and/or [REDACTED] employees, agents, or representatives.

8. WARRANTY.

[REDACTED] shall provide its Project and meet its obligations under this Contract in a timely and workmanlike manner, using knowledge and recommendations for performing the Project which meet generally acceptable standards in [REDACTED] community and region, and will provide a standard care equal to or superior to care used by service providers similar to [REDACTED] on similar Projects.

9. ENTIRE AGREEMENT.

This Contract contains the entire agreement of the parties, and there are no other promises or conditions in any other agreement whether oral or written concerning the subject matter hereunder. This Contract supersedes any prior written or oral agreements between the parties.

10. SEVERABILITY.

If any provision of this Contract will be held to be invalid or unenforceable for any reason, the remaining provisions will continue to be valid and enforceable. If a court finds that any provision of this Contract is invalid or unenforceable, but that by limiting such provision it would become valid and enforceable, then such provision will be deemed to be written, construed, and enforced as so limited.

We have carefully reviewed this contract and agree to and accept its terms and conditions. We are executing this Agreement as of the day and year first written above.

Reale Studios, LLC



By EDWARD SEEMAN

Name PARTNER

Title





By William H Page

Name Partner

Title

Exhibit B

Overview:

A re-implementation and significant improvement of the prototype WebbePhotoEditor, with full printing capability and serialization of files into various formats that include images, text, and formatting as explained below. The full functionality of the product is to be included in every distribution, however, only the book reading functionality will be enabled on installation. A one-time key is required to enable the other features, such as printing, creating, and editing. The exact features of that system are to be established with the contracted programming group.

Area	Specification	1.0	1.5	2.0
General				
	Functionality of wpe +	X		
	Simplified folder creation -- It is too hard to create a folder on the desktop.	X		
	Standard buttons for Bold, Italic, Underline, size and font and standard right click functionality	X		
	Cover with title insertion	X		
	Rotate photo 90°		X	
	2 languages			X
	Print big webbe		X	
	Insert photo or text on either side		X	
	Printing of various lengths of webbes		X	
	Process Wizard		X	
	Audio			X
	Translation engine			X
	Spell check			X
	Movie			X
	Hyperlink			X
	Assessment			X
	Graphics package			X
	Automatic and Prompt Modes		X	
	Right click menus		X	
	.rec File Creation / .use		X	
	.reb File Creation		X	
	.rep File Creation		X	
	Browse other projects in storage			X
	Pull down menus	X		
	Email Project		X	
	Full Edit Capabilities	X		
	Full "View" capabilities	X		
	Link to Website for Help	X		
	Full Help and FAQ		X	
	Upload to internet			X
	Download FROM internet			X
	Search Function			X
	Ability to insert a link			X

	Ability to insert TWO parallel languages			X
	Ability to insert a video			X
	Ability to have the book read aloud by the author, word by word or page by page.			X
Contact Sheet				
	Serializes and optimizes images in order		X	
	Image resizing and optimizing	X		
	Save, skip, rotate 90 degrees, etc.		X	
	Images available for use outside of the project	X		
	Reposition images	X		
	Delete images	X		
	Collect and Move groups of images to beginning or elsewhere in the contact sheet.		X	
	Blue Rectangle		X	
Split View				
	Move from contact sheet to Storyboard	X		
	Move groups of images to storyboard		X	
Storyboard				
	Control Characters X, V, C?	X		
	Text formatting: size, font, bold, italic, underline, color	X		
	Right click justification		X	
	Yoked pair movement	X		
	Yoked pair deletion and addition	X		
	Easy move text from one box to another with handle.		X	
	Images on left OR right, text on left OR right		X	
Book View				
	Displays images and text from storyboard as a book		X	
Printing				
	Sb5	X		
	Webbe	X		
	Contact sheet	X		
	PocketWebbe	X		
	Sb10	X		
	Print Setup		X	
	Printing any length webbe		X	

RealeBook Reader/Writer

Project Description

Overview:

The RealeBook Reader/Writer is a re-implementation and significant improvement of the prototype WebPhotoEditor, with full printing capability and serialization of files into various formats that include images, text, and formatting as explained below. The full functionality of the product is to be included in every distribution, however, only the book reading functionality will be *enabled* on installation. A one-time key is required to enable the other features, such as printing, creating, and editing. The exact features of that system are to be established with the contracted programming group.

File Formats:

- **Project:** (suffix: rep) When saved to this format, all the images and text are saved in a single file. When opened in the full-featured product, would result in an image contact sheet and storyboard being reproduced with the same formatting as when it was saved. If opened in the default version (reader only), the material from the Contact Sheet would be displayed as a non-editable book. We may want to have a prompt upon loading to show that acquisition of the full version would allow enhanced functionality.
- **Contact Sheet:** (suffix: rec) This format serializes all the optimized images and the order in which they are saved in the contact sheet. No Storyboard information is retained in this format. If opened in the default version, a message that this is viewable only in the full version is given.
- **Book:** (suffix: reb) This saves the images, text, and formatting from the Storyboard. When opened in the default mode, it opens as a book normally.

Views:

- **Project:** The default view is operational when creating a new project in the full featured product and it is the one that most users will use to compose and edit, unless they have a very low-resolution system. When opening a file saved as a project, this will be the default view. This view is split by a movable bar in the center that will allow the user to control the split between the RHS (contact sheet) and LHS (storyboard).
- **Storyboard:** Pulls the movable center bar all the way to the right, making the storyboard the only viewable portion.
- **Contact Sheet:** Pulls the movable center bar all the way to the left, making the contact sheet the only viewable portion.
- **Book:** This is the only view available in the default product. When opening a file saved as a book, this will be the default view.

Overarching Look and Description:

The product will be freely available and, without a license key, it can be used to read books that have been saved in the *reb* format by the **Writer**. The basic look for the **Reader**-only version should be similar to that shown in Figure 1 below. The tab for the **Writer** should lead to registration information leading to purchase and sharing of the full activation key. At a minimum, there should be information about who to contact and a website. To better control licensing, we would ideally also display a system unique number (hard drive ID or something unique to a given system). That number could be provided to generate a key to turn on the writer functionality. Also, that would be a good place to enter a registration key. The full version when **RealReader** is selected may be similar to this view, although the menu choices may be different. This security system must operate in both individual and site license conditions.

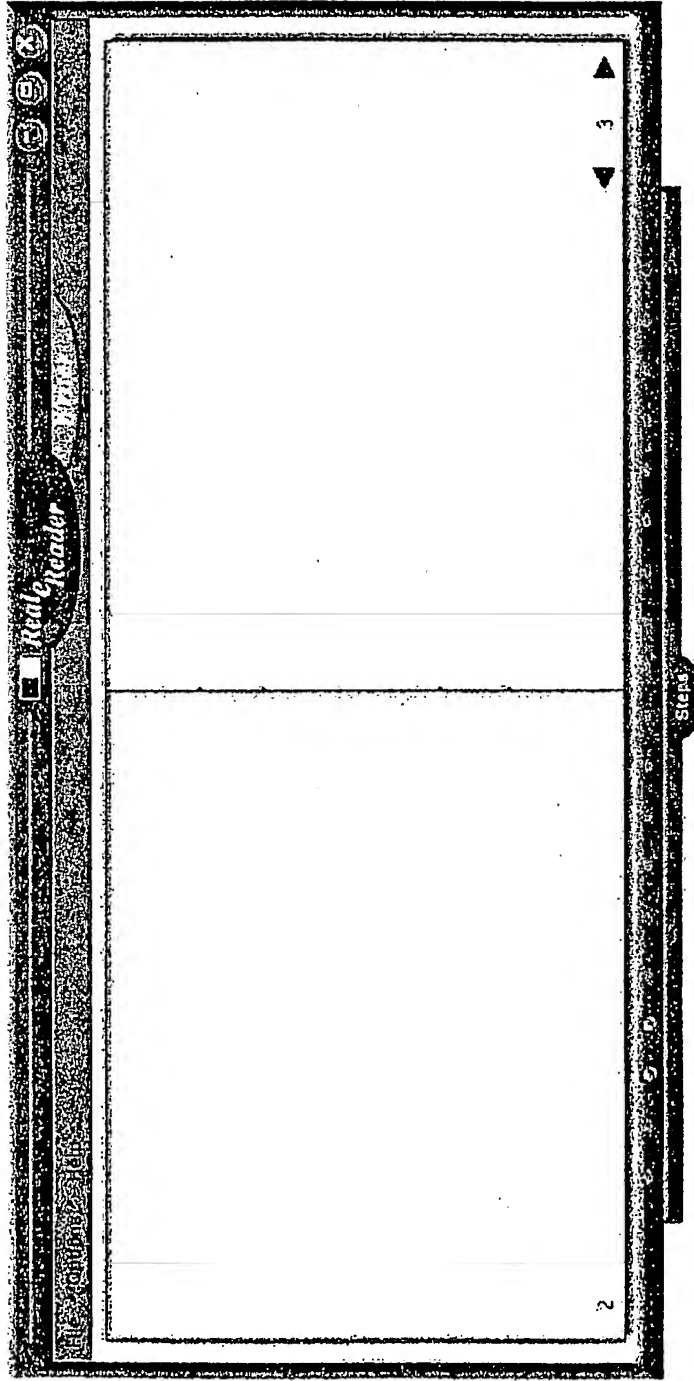


Figure 1: High-level look of reader

Once the product is registered, the **RealeWriter** tab view will be available. Selecting that tab will result in a structure similar to that shown in Figure 2. There are four on-screen views available. They are selectable from the menus or from tabs along the LHS. Depending upon which tab is selected, the content of the open area will change. The **RealeBook** view will allow a structure similar to that above, but with the capability of modifying the content of the book in the familiar book structure.

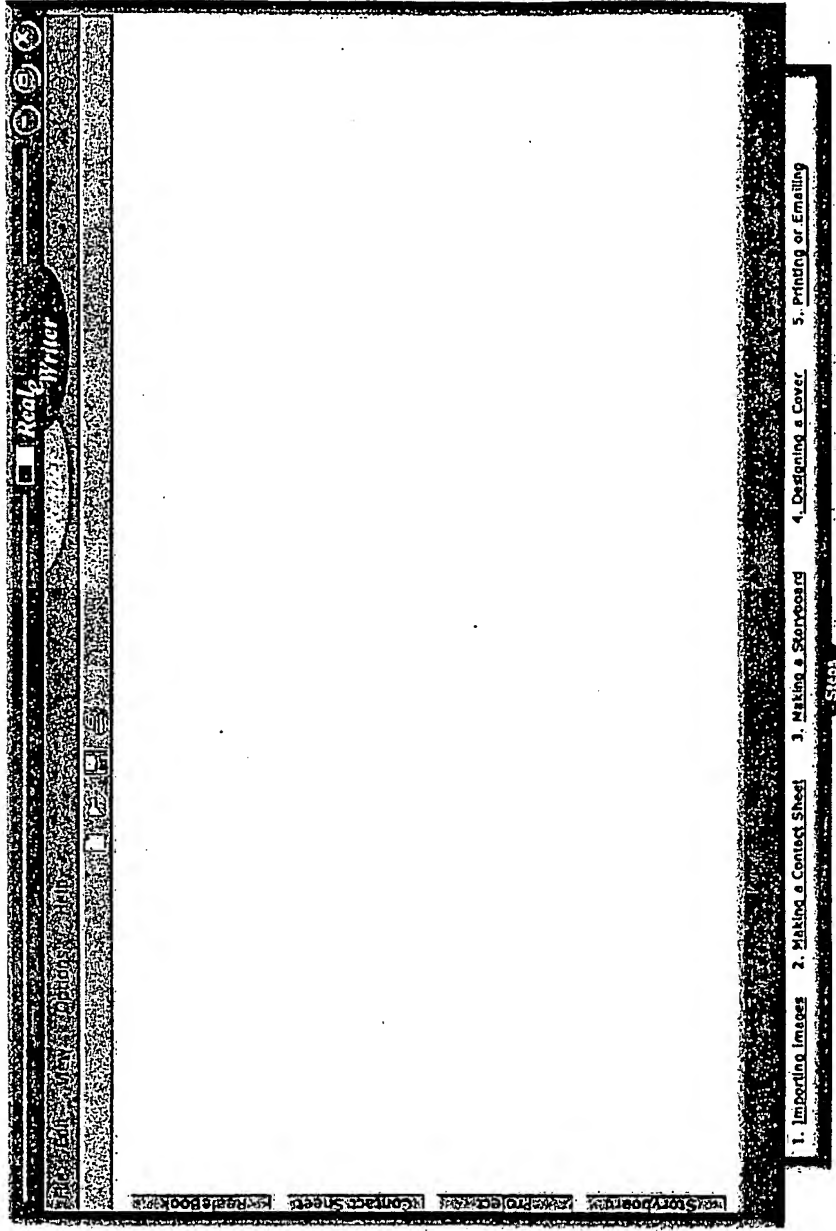


Figure 2: Writer overview for registered products, no onscreen view selected

Figure 3 is from a slightly older concept that has changed since it was developed. It is provided, however, to offer an overview of the other three possible views for the interior of the writer. The left side is the storyboard and the right side is the contact sheet. Primary functionality for each panel is described separately below.

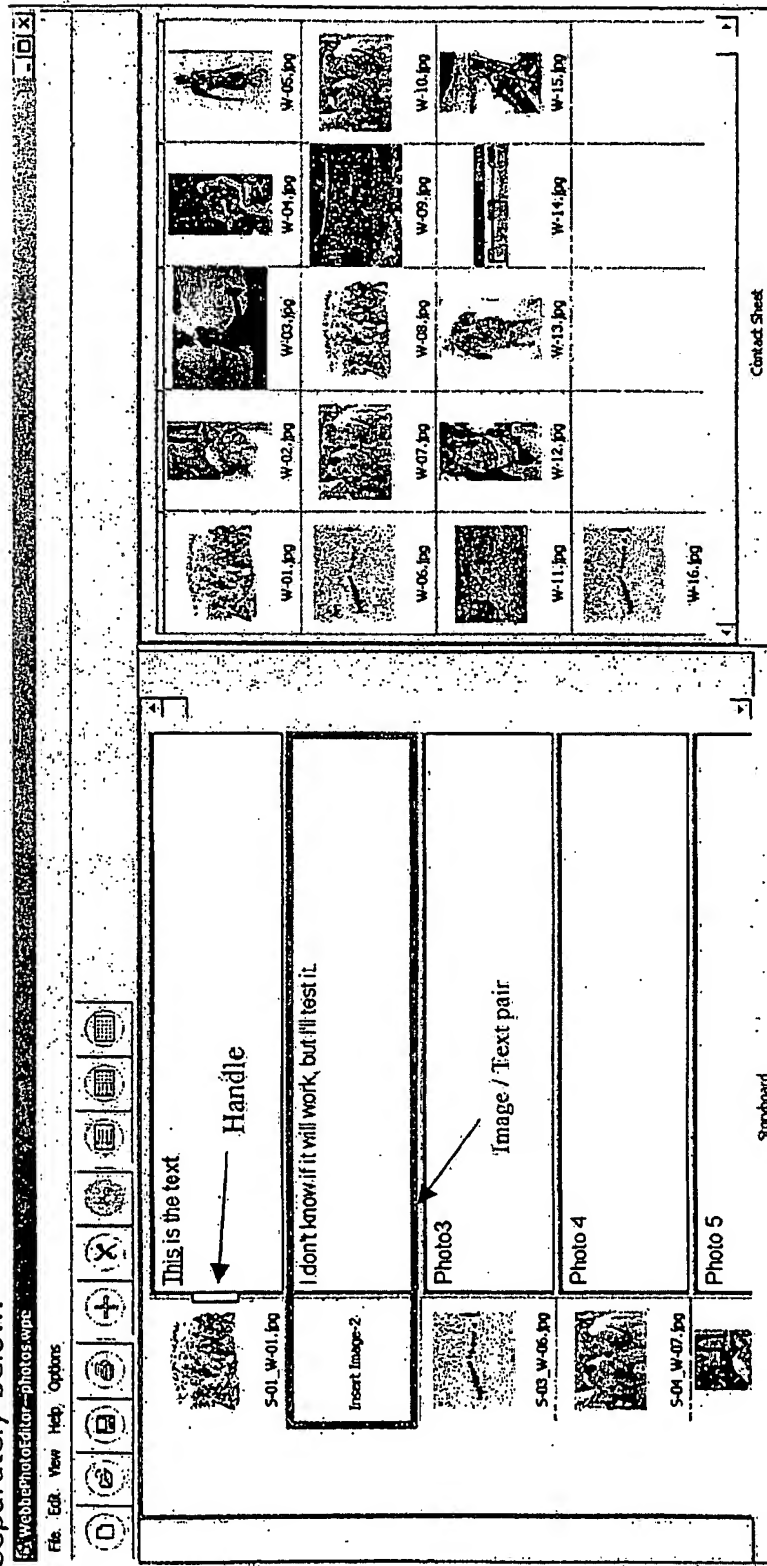


Figure 3: Old version overview – Used to exemplify the content of the registered Writer's Window

Contact Sheet

Create an interface through which individual images, or a folder that contains images (or subfolders of images) can be loaded. A folder of images from Windows Explorer, the Desktop, or any other location from which a folder may be grabbed will be dragged to the open space on the RHS (Contact Sheet shown in Figure 3 with pictures already added). Files in the folder are recursively checked to see if they are a type of graphic file (JPG, GIF, etc). Other file types are ignored. Each graphic file located is optimized and resized, with the resized image placed in a folder placed in the current working area. Moving an image would be accomplished by placing it on the image that is already in that spot and the program will drop every image one slot, with the replaced image ending up one slot down from where it was.

Image Conversion

The goal is to manipulate the images that are input into the system into optimized 4.44" x 3.33" images when printed, regardless of the shape, size, or resolution of the original image. That goal at 72 dpi, is a 320 x 240 pixel image. Under Edit → Options, the user can modify the image handling options to the Prompt setting described below, such as **Automatic** or **Prompt**.

Based on the selected options, resize at 72 dpi as follows:

1. Original image is exactly 4.44" by 3.33" in 72 dpi. Leave image unaltered in size.
2. If a proportional reduction will result in a 4.44" by 3.33"
 - a. **Automatic**: proportionally reduce to that size (default)
 - b. **Prompt** (as shown in Figure 4): show original image and prompt to proportionally reduce or crop. If crop is selected there are two possibilities, in preference order, only one of which need be implemented:
 - i. **Overlay**: Overlay a blue box on the image that can be moved and resized. After the users have placed the box on the part they want to be in the resulting image, they press

the enter key or click an on-screen button that says "crop". Using the parameters of the selected image portion only, apply the appropriate one of the methods to resize in automatic mode.

- ii. **Automatically Crop:** Crop from the top and bottom rows of pixels equal to $((\text{original height}) - 3.33") / 2$. Crop from each side of the image columns of pixels equal to $((\text{original width of image}) - 4.44") / 2$.
3. If the width of the original image is equal to or greater than 4.44" and the height of the image is less than 3.33"
 - a. **Automatic:** resize the image, so that the resized image is 4.44" in width, with the top and bottom equally filled with white pixels, making the overall image 4.44" by 3.33" (default).
 - b. **Prompt:** show original image and prompt to proportionally reduce or crop. If crop is selected there are two possibilities, in preference order, only one of which need be implemented:
 - i. **Overlay:** Overlay a blue box on the image that can be moved and resized. After the users have placed the box on the part they want to be in the resulting image, they press the enter key or click an on-screen button that says "crop". Using the parameters of the selected image portion only, apply the appropriate one of the methods to resize in automatic mode.
 - ii. **Automatically Crop:** Crop from each sides columns of pixels equal to $((\text{original width}) - 4.44") / 2$. Pad the top and bottom equally with white pixels so that the overall height is equal to 3.33"
4. If the height of the original image is equal to or greater than 3.33" and the width of the image is less than 4.44"
 - a. **Automatic:** resize the image, so that the resized image is 3.33" in height, with each side equally filled with white pixels, making the overall image 4.44" by 3.33" (default).
 - b. **Prompt:** show original image and prompt to proportionally reduce or crop. If crop is selected there are two possibilities, in preference order, only one of which need be implemented:
 - i. **Overlay:** Overlay a blue box on the image that can be moved and resized. After the users have placed the box on the part they want to be in the resulting image, they press the enter key or click an on-screen button that says "crop". Using the parameters of the

- selected image portion only, apply the appropriate one of the methods to resize in automatic mode.
- ii. **Automatically Crop:** Crop from the top and bottom row of pixels equal to ((original height) – 3.33”) / 2. Pad each side equally with white pixels so that the overall width is equal to 4.44”
5. If the height of an image is greater than 3.33” when the width of the original image is reduced to 4.44”
- a. **Automatic:** resize the image, so that the resized image is 3.33” in height, with each side equally filled with white pixels, making the overall image 4.44” by 3.33” (default).
 - b. **Prompt:** show original image and prompt to proportionally reduce or crop. If crop is selected there are two possibilities, in preference order, only one of which need be implemented:
 - i. **Overlay:** Overlay a blue box on the image that can be moved and resized. After the users have placed the box on the part they want to be in the resulting image, they press the enter key or click an on-screen button that says “crop”. Using the parameters of the selected image portion only, apply the appropriate one of the methods to resize in automatic mode.
 - ii. **Automatically Crop:** reduce proportionally, so the width is 4.44”. Crop from the top and bottom rows of pixels equal to ((original height) – 3.33”) / 2.
6. If the width of an image is greater than 4.44” pixels when an image is proportionally reduced to a height of 3.33” pixels
- a. **Automatic:** resize the image, so that the resized image is 4.44” in width, with the top and bottom equally filled with white pixels, making the overall image 4.44” by 3.33” (default).
 - b. **Prompt:** show original image and prompt to proportionally reduce or crop. If crop is selected there are two possibilities, in preference order, only one of which need be implemented:
 - i. **Overlay:** Overlay a blue box on the image that can be moved and resized. After the users have placed the box on the part they want to be in the resulting image, they press the enter key or click an on-screen button that says “crop”. Using the parameters of the selected image portion only, apply the appropriate one of the methods to resize in automatic mode.

- ii. **Automatically Crop:** reduce proportionally, so the height". Crop from each side columns of pixels equal to ((original width) - 4.44") / 2.
- 7. If width and height are both smaller than 4.44" and 3.33", respectively
 - a. **Automatic:** equally fill top and bottom with white pixels to an overall height of 3.33", and equally fill each side with white pixels to an overall width of 4.44". What if the small image we receive is 96 dpi or higher. Couldn't the image be just altered to 72 and THEN edited?
 - b. **Prompt:** user will be prompted to "pad or resize?" If "pad," equally fill top and bottom with white pixels to an overall height of 3.33", and equally fill each side with white pixels to an overall width of 4.44". If "resize," increase image as little as possible until the width 4.44" or height=3.33" and pad other dimension as necessary.

In all of the above, in the **prompt** mode, the user will be invited to **save, skip, rotate clockwise 90 degrees or rotate counterclockwise 90 degrees**. Other typical (but not exotic) image manipulation actions are also welcome, but not mandatory.

Optimized and modified images should be converted to **JPEG**, although they will be serialized into a single file, unless exported from the File Menu. These files may, however, be individually placed in a temporary folder for manipulation while a given project is open.

As each image is converted, a 1.33" by 1" version of the same image is displayed on the contact sheet. The **Contact Sheet** window should be sizable to a minimum that would allow a single column of images on the contact sheet. As the window is widened, as many images as will entirely fit should appear on each row. If there are too many images to entirely fit on contact sheet as displayed, a vertical scroll bar should allow navigation to the remaining images. The original filename prefix, followed by ".jpg", is displayed immediately under the thumbnail. Double-click will display the full sized resized / optimized image.

A right-click on any image will open a menu that includes **Save, Delete**, and **Properties**. **Save** will operate on the single image as though it had been selected and File → Export Images were chosen from the main menu. **Delete** will prompt for confirmation and then delete from the project. **Properties** will

open a dialog that allows you to provide a short description of the image or read one that was previously entered. This is different from the filename.

Images can be repositioned within the contact sheet by selecting and dragging with the mouse to any other position. A selected image would somehow show it is selected, and the target location selected would be apparent to the user.

A single image can be selected from the contact sheet with a left mouse-click. The image should be highlighted or bordered, such the selection is apparent. Multiple items can be selected, by holding down the ctrl key and clicking additional images. If multiple adjacent images are selected, then they can be all moved as a collection to anywhere on the contact sheet.

Double-clicking on a rec (contact sheet) file should launch this application, expand the images from the file and display the contact sheet as it was saved. Dragging a rec file onto an open and empty contact sheet will expand all the images from the file and display the images on the contact sheet as it was saved. Dragging a rec file onto a contact sheet that already contains images will operate the same as File → Append Contact Sheet.

Storyboarding and text formatting

The LHS is a structure that is commonly used to compose books. It is referred to as a **storyboard**.

Populate the LHS with placeholders for 10 image/text pairs as shown, displaying as many of the pairs as possible, beginning with the very top pair, and a vertical scroll bar to view the remaining image/text pairs. Toward the bottom of these image/text pairs will be a button that will add another blank image/text pair.

Images from the contact sheet can be “copy”-dragged to any position within the storyboard. The original will still remain on the contact sheet. Images can be dragged within the storyboard with the inserted image replacing the image in that position. If the “handle” (shown in Figure 3) is used to drag the item,

the image/text pair is moved to the new position (or immediately before the pair, if dragged onto an existing pair), with all subsequent pairs being moved down, filling the space vacated by the moved pair.

Text can be added to any of the storyboard text boxes, leaving a minimum of approximately 1/8" between the text and the nearest border. Text will be vertically centered and left justified as a default. As lines are added, the text should remain vertically centered as a default. As text exceeds the size of the text area, a scroll bar appears. The cursor stays visible and the section above scrolls up under the text area directly above it. When the text area is no longer active, it returns to show the very first line of the text.

Right clicking on a text box will give the option to justify vertically or horizontally. Multiple text boxes can be selected before right-clicking. In that case, justification will be applied to all selected boxes. Additionally, the user should be able to select to apply the formatting to all boxes in this document and/or make this the system default for all future text additions.

Text can be selected and can be moved between text boxes or between this application and other Windows application with the common edit techniques, such as highlight-drag-drop, **ctrl-c**, **ctrl-x**, **ctrl-v**, and edit menu selections. Font formatting moves with the text. If an entire paragraph (which may be a single line) or the entire text area is selected and moved, paragraph formatting is retained with the text.

The default font is 14 point Comic Sans, however, the default can be changed from the preferences panel through Edit → Preferences. When that change is made, it is also applied to all text cells that do not yet have any text in them.

Any part of the text can be formatted, with **face**, **point size**, **bold**, **italic**, **underline**, and **justification**, using common techniques (such as double-clicking or dragging across the text), then right-clicking on the selected text, choosing the appropriate selection from the Edit menu, or from the toolbar.

Moving in the Storyboard:

Between the image and text box, there should be a small but evident vertical symbol called the "**handle**". If a handle is dragged to another location in the storyboard, both the image and the text are moved to the new location, with all subsequent image/text pairs moving down one row. Meanwhile, the place from which the image/text pair was taken closes up and all subsequent image/text pairs are moved up one row. If the handle is dragged off the storyboard, the pair is deleted, moving all subsequent pairs up one position.

Right-clicking on the handle also gives the option of deleting the pair, moving all subsequent pairs up one position, or deleting the pair and leaving an empty image/text pair in the space.

If an image alone is selected, it may be dragged to any other image location, replacing the image that is currently there. There should be some indication like change of image frame indicating that an image is selected and there should be a clear indication what location for the move has been selected.

Each text box should also have a **handle in the corner** that will allow it to be dragged to any other text box, replacing all text that is currently there.

Right-clicking on the handle also offers the option to insert a blank pair directly above at that location, moving that image/text pair and all subsequent pairs down one position.



Figure 4: Prompt mode selection of target portion of original image

Menus

File → Export Images will prompt for a location and directory name. A directory with that name in the specified location will be created. If no images on the contact sheet are selected, each optimized file will be copied from the working directory to the specified directory. If images are selected, only those selected will be copied. That dialogue box offers the option to create a new folder if desired.

File → Append Contact Sheet will prompt for an existing rec file. Once selected, those images are appended to the existing open project and contact sheet.

File → Review will allow browsing through directories that include rec files. The list of files will be displayed, along with the number of photos contained and the description (if any) that was entered for that contact sheet. Users can then click on the description and all of the images are displayed with their descriptions and file sizes. Clicking on one of those image descriptions invites saving the images somewhere else.

File → E-mail is available if the user has configured e-mail settings and the user is currently online. If the storyboard has been changed since the last time it was saved, the user will be prompted to save, only if he elects to save, will he be shown the e-mail dialog. The dialog will allow one or more e-mail addresses to be entered. Additionally, there should be a text box for a "personalized message" and the ability to select Project, Contact Sheet, or Book, with book as the default. When a "send" button is clicked, an e-mail is generated and sent through a pre-configured SMTP server to each of the recipient e-mail addresses. The e-mail will include the personalized message and the selected file type as an attachment.

File → Print brings up a print dialog. The contact sheet will be printed with the option of having the file names below each image or not. The numbers of images on a single printed page will be determined by the file name choice and the established printable area.

Each page is numbered with the page number and the total number of pages printed, such as "2 of 5" for the second page of a five page contact sheet. If a description has been added to the contact sheet, it should appear at the top left of the page. If the file has been saved, its filename should appear at the top right of the page.

File → Print Setup allows a user to determine margins for the current document and toggles to omit the display of the description, filename, and page numbering. The print setup is retained with a given project, however, the same setup screen should also have a button to set this as the **default** for all newly created contact sheet projects.

File → **Print Setup** should allow a user to determine **margins** for the current document and toggles to **omit the display of the description, filename, and page numbering**. The print setup is retained with a given project; however, the same setup screen should also have a button to set this as the default for all newly created storyboard projects.

File → **Save** will prompt for a filename and location and will serialize all image files and text in a single file with an extension of rep. Additionally, there should be a location for a short description about the project. This description will be written in a scroll box and that file will travel with the rep file.

File → **Save Contact Sheet** will prompt for a filename and location and will serialize all image files in that single file with an extension of rec. Additionally, there should be a location for a short description about the contact sheet.

When either: 1. a new file is opened, 2. the application is closed, or 3. an application is first opened, all temporary working directories and contained images are deleted. If the files to be deleted have not yet been saved as a rep file, the user should be prompted with the option to do so.

File → **Append Storyboard** will prompt for an existing rep file. Once selected, those image/text pairs are expanded and appended to the existing open storyboard. The images from the storyboard will be appended to the contact sheet (Project 1) as well.

File → **Properties** gives access to all the configuration options.

Edit → **Undo** text entering or changes in image locations. The undo will offer a string of undos from which the user can select backward one at a time.

Edit → **Cut** selected image, text, or an image/text pair if the handle was selected.

Edit → **Copy** as above

Edit → Paste

Edit → Font for highlighted text, or beginning from the current cursor location.

View → Project – selects the split view of half contact sheet and half storyboard. **Note:** That tab in Figure 2 that says Split View should read **Project View**

View → Storyboard

View → Contact Sheet

View → RealeBook

Help → On-line Help and FAQ

Help → Update or Register

Help → About...

Toolbar

The toolbar will allow for one-touch operation of many of the items available from the menus as well as other functions.

Printing

File → Print or clicking on a print icon on the toolbar will bring up a print dialog from which you may choose **Webbe, Pocket Webbe, Contact Sheet, SB5 (Storyboard), or SB10 formats** and the pages to print indicates the resulting total pages of that particular choice, with “all” as the default.

For the Contact Sheet or either of the SB formats, the user may select to print any ONE page, or a series of non-contiguous pages.

Note: The **Webbe** and **PocketWebbe** are actual books. They need covers with both images and text on them. That will likely necessitate a dialogue box in printing asking for the image and title, or some other provision for the user to enter that data. Note the **STEPS** tab at the bottom of the **RealeWriter** in **Figure 2**.

Webbe This will compute the proper layout for a small format book (sample to be provided) and will provide progressive instructions to the user, explaining the process. This process will differ, depending upon the length of the book. This print format will allow printing from one sheet of paper to four sheets of paper, front and back.

PocketWebbe This will compute the proper layout for a small format book (sample to be provided) and will provide progressive instructions to the user, explaining the process. This format is designed to print on two sides of a single sheet of paper.

Contact Sheet prints the contact sheet, with as many 1.33" x 1" images and their filenames in each row as the margins will allow.

NOTE: Figure 5: Storyboard 5 and Figure 6: Storyboard 10 follow this page.

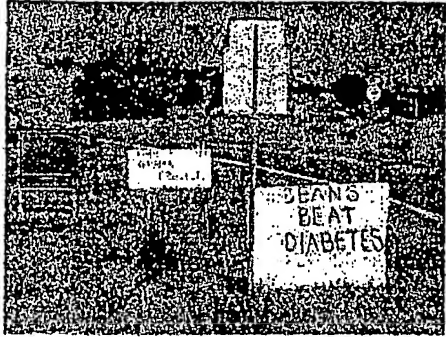
SB5 prints five image /text pairs on a page as shown in **Figure 5**.

SB10 prints ten image/text pairs on a page. The images should be as large as possible to allow that number of images on a page, with the text area the same height and consuming the remaining area to the right margin. Each image/text pair should be surrounded by a box. If there is too much text to fit into the box, the remaining an ellipsis should be added to the end of the text and the remaining text should be omitted from the printing, leaving the .wsb file intact with whatever text was entered in

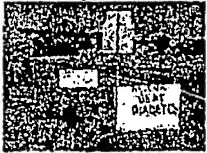
the first place. There should be some kind of prompt that indicates that this ellipses deal is going to be used and that if the user would like to reduce the fontsize so that all the text is readable, then they should hit CANCEL and do that before trying to print. The guide would be that the text that fits into the text area without generating a scroll bar would all be printed.

Name:

Date:



Enter text here. As the text gets longer, it wraps, but stays in the middle. If the text is too long for the text area, the print function prints the first X visible lines and truncates the rest...see SB 10 below.

	Enter text here. As the text gets longer, it wraps, but stays in the middle. If the text is too long for the text area, the print function prints the first X visible lines and
2	The size of the cells on the storyboards should be constrained so that SB5 prints five rows and SB10 prints 10 rows filling a single piece of paper.
4	
6	
8	
10	
12	
14	

Each page is numbered with the page number and the total number of pages printed, such as "2 of 5" for the second page of a five page storyboard. If a description has been added to the storyboard, it should appear at the top left of the page. If the file has been saved, its filename should appear at the top right of the page.

Double-clicking on a **rep** file should launch this application, expand the images from the file and display the contact sheet as it was saved. Dragging a **rep** or **reb** file onto the open and empty writer will expand all the image/text pairs from the file and display them on the contact sheet as it was saved. Dragging a **reb** or **rec** file onto a contact sheet that already contains images will operate the same as File → Append Contact Sheet.

File → Review will allow browsing through directories that include **wsb** files. The list of files will be displayed, along with the number of image/text pairs contained. And the descriptions for each wsb file.

On-line interactions

We would like for this project to interact with an on-line project that is currently under development. Because that product is in flux and we don't have the exact functional specs for it, the initial version should be able to **save any or all of the five print views into an HTML format** and then be able to **FTP** from the application to a server would be sufficient.